

明 細 書

セキュリティホール診断システム



5 技術分野

本発明は、コンピュータのセキュリティホールの有無を診断するシステムに関するものである。

背景技術

- 10 図9は特開2001-337919（第4-8頁、図3、図4、図14）に代表される、従来のセキュリティホール診断システムを示す構成図である。従来のシステムは操作装置900と検査実行装置907で構成され、操作装置900はディスプレイ902、画面生成部903、操作制御部905、表示名定義ファイル904及び手順定義ファイル906で構成されている。

また、検査実行装置907は、実行制御部908、対象ホスト情報格納部909、複数の検査実行部911、及び検査実行手段格納部910で構成されている。

- 20 図10は同システムにおける手順定義ファイル906の例を示すものである。手順定義ファイル906には、検査実行手段911の分類キー名と、分類キーとして指定された検査実行手段911のプロパティの値毎に表示名、実行タイプ、説明文が記載されている。

- 25 図11は同システムにおける検査実行手段911の情報（検査実行情報）を表すものである。検査実行情報には、各検査実行手段911を特徴付ける値（プロパティ）がキー名（プロパティ名）に関連付けられて格納されている。つまり、検査実行情報（検査実行手段の情報）は、検

査実行手段に一つずつ含まれており、その検査実行手段を特徴付ける情報（＝プロフィール）である。検査実行情報には、複数の項目（プロパティ）を記述することができる。各項目は、プロパティ名で区別される。

- 5 次に従来のシステムの動作について説明する。操作装置 9 0 0 が検査実行装置 9 0 7 に接続されると、操作装置 9 0 0 は表示名定義ファイル 9 0 4 及び手順定義ファイル 9 0 6 をロードする。

- 10 次に、検査実行装置 9 0 7 中の検査実行手段格納部 9 1 0 に蓄積されている検査実行手段 9 1 1 一つ一つから検査実行情報を取り出し、手順定義ファイル 9 0 6 に指定されたキー名に対応するプロパティをもとに、各検査実行手段 9 1 1 を手順定義ファイル 9 0 6 記載のカテゴリに分類する。最後に分類された検査実行手段 9 1 1 の一覧をカテゴリ毎にディスプレイ 9 0 2 に表示する。

- 15 使用者 1 0 1 はディスプレイ 9 0 2 に表示されたカテゴリを選択し、実行に必要なパラメータを入力し、検査実行を要求する。パラメータの説明は表示名定義ファイル 9 0 4 に記載されている情報が利用される。検査実行を要求された操作装置 9 0 0 は、そのカテゴリに分類された検査実行手段 9 1 1 を実行するように、操作制御部 9 0 5 を通じて検査実行装置 9 0 7 に要求する。

- 20 検査実行装置 9 0 7 は指定された検査実行手段 9 1 1 を呼び出し、その結果、検査のためのパケットが検査対象ホストコンピュータ 1 0 7 に送信される。

- 25 なお、各検査実行手段 9 1 1 は、対象ホスト情報格納部 9 0 9 に情報を格納することができ、格納された情報は、他の検査実行手段 9 1 1 によって参照することができる。また、使用者 1 0 1 が操作装置 9 0 0 を通じて直接対象ホスト情報格納部 9 0 9 に情報を格納することも可能で

ある。

以上が従来のシステムにおける検査の流れである。ここで、カテゴリの表示順序は、手順定義ファイル 9 0 6 に記載されている順序であり、これを一般的な攻撃の手順に沿うようにすることで、使用者 1 0 1 はディスプレイ 9 0 2 に表示された順に検査を行うことによって攻撃者を模
5 擬した検査を行うことができる。

以上のように、従来のセキュリティホール診断システムは、複数の検査実行手段を有し、それらを手順定義ファイルで与えられた方法で分類・表示し、カテゴリ毎に使用者が選択することでそのカテゴリに属する
10 検査実行手段を実行するというものであり、また検査実行手段は直接検査対象ホストコンピュータに対し検査を実行するというものであった。そのため、以下のような問題があった。

各カテゴリ毎に入力しなければならない実行パラメータは、使用者が前の検査の結果から入力しなければならず、あるカテゴリの検査の結果
15 と次のカテゴリへの入力との関係を使用者は理解している必要がある。そのために、使用者はセキュリティ上の知識を必要とされた。

定義ファイルは順次実行のシナリオしか表現できないが、実際の攻撃者は、前に行った攻撃の結果に応じて次に実施すべき攻撃の種類を変化させる場合も多い。従来のシステムでは、次にどのカテゴリの検査を実
20 行するか判断は使用者が行わなければならず、ここでも使用者にセキュリティ上の知識を必要とした。

攻撃者は、ある目的を持って複雑なステップで構成される攻撃を行う。その一連の攻撃は、さらに大きな目的を達成するための攻撃シナリオの一ステップに過ぎない場合も想定される。従来のシステムでは、この
25 ような階層化された攻撃シナリオを表現することはできない。

対象ホスト情報格納部に蓄積される情報から、別の情報を推論するた

めの推論手段がない。これは、例えば対象ホストのOSがUNIX（登録商標）であることから管理者のアカウント名がrootであるという知識を導出するための手段である。従って、各検査実行手段には、必要とする情報を、蓄積されている情報から推論するためのロジックを埋め込まなければならない。

攻撃者はあるホストへの侵入に成功すると、そこを踏み台としてさらに内部への侵入を試みる場合が多い。しかし従来の検査システムでは検査実行手段から直接検査を行うため、踏み台の使用を用いた検査シナリオを実施できない。

本発明は上記の問題を解決するためになされたものであり、以下の事柄を目的とする。

検査シナリオを、プログラミング言語で記述されたスクリプトとして表現し、スクリプトから自動的にプラグイン（検査実行手段に該当）を呼び出すことで、複雑な試験の実施を可能とする。

各検査実行手段間のパラメータの授受はスクリプトが媒介するようにすることで、使用者は検査実行手段間の入出力の関係について知っている必要を無くす。

セキュリティホール診断を行う際に、より現実に近い高度な攻撃シナリオに基づいた検査を実施することを可能とし、使用者に必要とされるセキュリティの知識の程度を軽減することができ、検査ロジックの作成者の負担を軽減する。

発明の開示

本発明に係るセキュリティホール診断システムは、不正アクセスのために通常攻撃者が行う手順がプログラミング言語で記述されたスクリプトが複数蓄積されたスクリプト蓄積部と、

- 利用者からの入力により上記スクリプトの一覧を要求する操作部と、
上記操作部の要求に応じ、上記スクリプト蓄積部から各スクリプトを
取り出し、入出力パラメータ記述、スクリプト実行必要条件、検査手順
を表示したリストを作成して利用者に提示し、利用者が選択したスクリ
5 プトを実行するスクリプト制御部と、
個々のセキュリティホール攻撃のためのロジックが実装されたプラグ
インが蓄積されたプラグイン蓄積部と、
スクリプト制御部がスクリプトを実行することにより呼び出され、上
記プラグイン蓄積部から上記実行スクリプトに対応するプラグインを取
10 り出して、そのプラグインを検査対象コンピュータに対して実行するプ
ラグイン制御部
とを備えた。

図面の簡単な説明

- 15 図 1 は、実施の形態 1 に係るセキュリティホール診断システムの概略
構成図。
図 2 は、図 1 に示す脆弱性検査装置の内部構成図。
図 3 は、図 1 に示す踏み台模擬プログラムの内部構成図。
図 4 は、スクリプトの構成説明図。
20 図 5 は、スクリプト制御部の動作流れ図。
図 6 は、クラス名を指定して検査を実行する場合の動作流れ図。
図 7 は、知識ファイルの例を示す説明図。
図 8 は、スクリプトの記述例を示す説明図。
図 9 は、従来のセキュリティホール診断システムを示す構成図。
25 図 10 は、従来のシステムにおける手順定義ファイルの説明図。
図 11 は、従来のシステムにおける検査実行部の情報（検査実行情報

) の説明図。

発明を実施するための最良の形態

実施の形態 1.

5 はじめに図 1 を参照しながら、本システムの概要について述べる。本システムはローカルで動作する脆弱性検査装置 100 とリモートもしくはローカルのホストコンピュータである 1 つ以上の踏み台模擬装置から構成される。本実施の形態においては 1050、1060 の二つの踏み台模擬装置が配置され、脆弱性検査装置 100 と踏み台模擬装置 1050、1060 はネットワークで接続されている。また、踏み台模擬装置 1050、1060 は、それぞれ踏み台模擬プログラム 105、106 を実行する。

脆弱性検査装置 100 は、使用者 101 からの要求に応じて、対象となるホストコンピュータ又は、ネットワークに対してセキュリティ上の脆弱性があるかどうかを検査する計算機である。検査は脆弱性検査装置 100 が踏み台模擬装置 1050 の踏み台模擬プログラム 105 を操作することで実施される。

20 踏み台模擬装置 1050 が実行する踏み台模擬プログラム 105 はネットワークを通じて脆弱性検査装置 100 から命令を受け取り、パケット送受信、プロセスの起動・終了、ファイル転送、メッセージ中継を行うプログラムである。

25 踏み台模擬プログラム 105 は他の踏み台模擬装置 1060 の踏み台模擬プログラム 106 に命令を転送する機能も有しており、踏み台模擬装置 1050、1060 を適切に配置することで内部ネットワークに位置する検査対象ホストコンピュータ 107 に対しても検査が行えるようになる。

踏み台模擬プログラム105、106は、検査前に検査対象のネットワーク上のホスト内で、動作させておくことも、また、脆弱性検査の一環として、セキュリティホールを利用して埋め込むことも可能である。

5 踏み台模擬プログラム105の操作は、実際には脆弱性検査装置100内でプラグイン104により制御される。プラグイン104とは、個々のセキュリティホールを攻撃するための動的ロード可能な共有ライブラリである。プラグイン104は踏み台模擬プログラム105を操作することで検査対象上に存在するセキュリティホールへの攻撃を行う。

10 様々なプラグイン104を用意することで、多様なセキュリティホールに対する脆弱性検査が可能となる。

プラグイン104はスクリプト102によって制御される。スクリプト102とは、不正アクセスのために通常攻撃者が行う手順をインタプリタ言語で記述したテキストデータである。スクリプト102に従って様々なプラグイン104を呼び出すことで、脆弱性検査装置100は、
15 攻撃者を模擬した複雑な脆弱性検査を行うことが可能となっている。

スクリプト102もプラグイン104同様、その目的に応じて複数用意することができる。また、スクリプト102から他のスクリプト102を呼び出すことも可能であり、これにより他のスクリプト102を攻撃の一ステップとするような、より高度なスクリプト102を記述する
20 ことができる。

本実施の形態では、スクリプト102の記述言語としてPerlを使用している。

スクリプト102は、検査を実行した結果得られた検査対象に関する知識、例えばユーザアカウントの一覧や動作しているサーバー一覧等の情報
25 報を、知識共有部103に蓄えることが可能である。知識共有部103に蓄積された知識は、他のスクリプト102から参照することができる

また、知識共有部 103 に推論ルールに基づいて知識を吟味する推論部 108 を備えることで、スクリプト 102 から得られた知識（事実情報）から新しい知識（推論）を導出することも可能である。例えばあるスクリプト 102 によって検査対象ホストコンピュータ 107 の OS が UNIX（登録商標）系であることが判れば、推論ルールにより、そのホストの管理者アカウント名が root である、という知識を導出することができる。

以上の概要を踏まえた上で、次に、図 2 を参照しながら脆弱性検査装置 100 の内部構成について説明する。脆弱性検査装置 100 は、操作部 201、検査実行部 202 で構成されており、検査実行部 202 はスクリプト制御部 203、プラグイン制御部 204、知識共有部 103、及び踏み台模擬プログラム制御部 205 で構成されている。

スクリプト制御部 203 は、スクリプト 102 を蓄積・閲覧・実行するための手段を提供する。一つ以上のスクリプト 102 がスクリプト制御部 203 内にあるスクリプト蓄積部 206 に蓄積されている。スクリプト蓄積部 206 内でスクリプト 102 は、ファイル名によって一意に名前付けされ管理されている。また、スクリプト蓄積部 206 は、たとえば磁気ディスクである。

スクリプト 102 は、図 4 に示すとおり、クラス名記述部 401、実行条件記述部 402、入出力パラメータ記述部 403、説明記述部 404、及び検査手順記述部 405 で構成されている。

クラス名記述部 401 には、そのスクリプト 102 がどのようなカテゴリの検査に属しているかを表すデータが記述されている。実行条件記述部 402 には、スクリプト実行時に満たされていなければならない条件が記述されている。条件は述語論理を用いて記述される。入出力パラ

メータ記述部 403 には、スクリプト 102 がどのような入力を受け取り、どのような出力を行うかが記述されている。説明記述部 404 には、スクリプト 102 の説明文が記述されている。検査手順記述部 405 には検査手順が記述されている。

- 5 スクリプト 102 の記述例を図 8 に示した。図中、"Class：" がクラス名記述部 401 を表し、"Precondition："が実行条件記述部 402 を表し、"Input："及び"Output："が入出力パラメータ記述部 403 を表している。"Description："が説明記述部 404 であり、"#-----END__SCRIPT__PROPERTY-----"より下の部分に検査手順記述部
10 405 となる Perl コードが記載される。

- プラグイン制御部 204 内にはプラグイン蓄積部 207 があり、1つ以上のプラグイン 104 が蓄積されている。プラグイン蓄積部 207 は、たとえば磁気ディスクである。プラグイン 104 は、プラグイン蓄積
15 部 207 内で一意に名前付けされて管理されている。

 知識共有部 103 は、スクリプト 102 が脆弱性検査の過程で収集した知識を他のスクリプト 102 と共有することを可能とするための手段である。

- 知識共有部 103 内には、知識蓄積部 208 があり、脆弱性検査の過程で収集された知識が蓄積されている。知識蓄積部 208 は、たとえば
20 磁気ディスクである。また、知識共有部 103 内には推論部 108 があり、知識蓄積部 103 内の知識を元に推論処理を行うことが可能となっている。推論処理の一環としてスクリプト制御部 203 を通じてスクリプト 102 を実行することも可能である。

- 25 踏み台模擬プログラム制御部 205 は、プラグイン 104 に対し踏み台模擬プログラム 105 を制御するためのインタフェースを提供すると

ともに、稼動中の踏み台模擬プログラム 1 0 5 の状態管理も行う。

なお、脆弱性検査装置 1 0 0 は、例えばマイクロプロセッサ等の CPU、半導体メモリ等や磁気ディスク等の記録手段、及び通信手段を有する計算機により実現することができる。図 2 に示した知識共有部 1 0 3、スクリプト制御部 2 0 3、プラグイン制御部 2 0 4 及び踏み台模擬プログラム制御部 2 0 5 をプログラム（脆弱性検査プログラム）とし、記録手段に脆弱性検査プログラムを格納し、CPU が脆弱性検査プログラムを読み込むことにより脆弱性検査装置 1 0 0 の動作を制御し、以下に示す処理を実現するようにしてもよい。

次に、図 3 を参照しながら図 1 中の踏み台模擬装置 1 0 5 0 が実行する踏み台模擬プログラム 1 0 5 の内部構成について説明する。踏み台模擬プログラム 1 0 5 は、全体制御部 3 0 1、通信中継部 3 0 2、検査パッケージ送受信部 3 0 3、プロセス実行部 3 0 4 及びファイル転送部 3 0 5 で構成されている。通信中継部 3 0 2 は、ネットワークを通じて、他の踏み台模擬装置 1 0 6 0 の踏み台模擬プログラム 1 0 6 や図 2 に示す踏み台模擬プログラム制御部 2 0 5 と通信を行う。

全体制御部 3 0 1 は、通信中継部 3 0 2 を通じて送られてきた制御メッセージを受け取り、その指示に従って検査パッケージ送受信部 3 0 3、プロセス実行部 3 0 4、ファイル転送部 3 0 5 を操作する。また、制御メッセージが自分宛で無い場合には通信中継部 3 0 2 を利用して、制御メッセージを本当の宛先に転送する。

通信中継部 3 0 2 は制御メッセージを転送する。通信中継部 3 0 2 は、一つの親と複数の子と接続可能である。そのため、踏み台模擬装置 1 0 5 0 は、脆弱性検査装置 1 0 0 を頂点としたツリー状に相互接続される。

接続は、TCP によって行われ、TCP 接続要求は子から親、親から

子どちらからも可能である。

次に図 2 を用いて本システムの動作について説明する。

はじめに使用者 1 0 1 は操作部 2 0 1 を通じて、検査実行部 2 0 2 に対し、実行可能なスクリプト 1 0 2 の一覧を要求する。検査実行部 2 0 2 はその内部手段であるスクリプト制御部 2 0 3 を呼び出す。

スクリプト制御部 2 0 3 は、スクリプト蓄積部 2 0 6 からスクリプト 1 0 2 を一つずつ取り出し、そのファイル名、入出力パラメータ部 4 0 3、説明記述部 4 0 4、及びクラス名記述部 4 0 1 の内容をリストに蓄積する。全てのスクリプト 1 0 2 に対してこの処理を繰り返したら、リストを操作部 2 0 1 を通じて使用者 1 0 1 に返す。

次に、使用者 1 0 1 は検査の一覧（リスト）から自分の行いたいスクリプト 1 0 2 を選択し、操作部 2 0 1 を通じて検査実行部 2 0 2 に対し、検査の実行を要求する。要求には、（１） スクリプト名もしくはクラス名、（２） 検査パラメータの情報、（３） 検査終了条件（但し（１）がクラス名の場合のみ）が含まれている。検査実行部 2 0 2 は、スクリプト制御部 2 0 3 に対し、検査の実行を要求する。実行結果は操作部 2 0 1 に返される。

次に図 2、図 4、図 5 を参照しながらスクリプト制御部 2 0 3 の動作について説明する。はじめに検査名を指定して検査を実行する場合について説明する。

検査実行要求を受けたスクリプト制御部 2 0 3 は、ステップ 5 0 1 でスクリプト蓄積部 2 0 6 内に指定されたファイル名で管理されたスクリプト 1 0 2 を取り出す。

次にステップ 5 0 2 で、スクリプト制御部 2 0 3 はスクリプト 1 0 2 に記載されている実行条件記述部 4 0 2 の内容を取り出す。スクリプト 1 0 2 の実行条件記述部 4 0 2 には、そのスクリプト 1 0 2 を実行する

ために必要な条件、例えば検査対象ホストコンピュータ107のOSがWindows（登録商標）であること等が述語論理で記述されている。スクリプト制御部203は、この条件を知識共有部103に渡し、実行条件が満たされているかどうかを確認する。

- 5 次に知識共有部103からの応答を元に、実行条件が満たされたかどうかの判断をステップ503で行い、もし実行条件が満たされていないればスクリプト制御部203は、ステップ508に進みスクリプト102の実行失敗として処理を終了する。

- 10 もし実行条件が満たされていたならば、処理はステップ504に進む。ここでスクリプト制御部203は、スクリプト102の検査手順記述部405の内容と、検査実行要求に含まれる検査パラメータに従い、検査を実行する。

ステップ505でスクリプトの実行結果が判断され、失敗した場合はステップ508に進み、処理を終了する。

- 15 実行に成功した場合、新たな知識が獲得される場合がある。例えば、発見されたセキュリティホールの一覧等である。そのような知識は他の検査を行うときに再利用できるようステップ506で、知識共有部103中の共有知識蓄積部208中に格納しておく。

- 20 最後に、実行結果を呼び出し元に返して処理は終了する（ステップ507）

次に、図6を参照しながらクラス名を指定して検査を実行する場合について説明する。

- 25 検査実行要求を受けたスクリプト制御部203は、ステップ601～ステップ607で構成されるループを実行することで、スクリプト蓄積部206中に格納されているスクリプト102を順に取り出し、以下の動作を行う。

まず、ステップ604で現在対象としているスクリプト102のクラス名記述部401を参照し、そのスクリプト102が検査実行要求で指定されたクラスに所属しているかどうかを検査する。

もしスクリプト102が検査実行要求で指定されたクラス102に所属していなければ、ステップ609に進み、次のスクリプト102に対して、処理を行う。

スクリプト102が検査実行要求で指定されたクラスに所属していたならば、ステップ605で、スクリプト102の実行を試みる。具体的には、図5のステップ502からの処理を行うことになる。

10 ステップ606で実行の成功・失敗を判断し、もし失敗したならば、ステップ609に進み、他のスクリプト102の実行を試みる。

実行が成功した場合、さらに他の同一クラスのスクリプト102を実行するかどうかをステップ607で判断する。判断は、検査実行要求として渡される情報に含まれる、検査終了条件をもとに行われる。

15 もし、検査終了条件が、「クラスが一致する全てのスクリプトを実行」であったならば、ステップ609に進み、他のスクリプト102についても実行を試みる。そうでなければステップ608に進み、実行結果を呼び出し元に返して処理は終了する。

20 ステップ602で、全てのスクリプト102に対して実行を試みたかどうか判定され、もし全てのスクリプト102に対して実行を試みたことが判明した場合には、処理はステップ610に進む。

ステップ610に到達するまでに、一つでもスクリプト102の実行に成功していた場合には、ステップ608に進み、実行結果を呼び出し元に返して処理は終了する。もし一つも成功していなかった場合には、
25 ステップ611に進み、検査実行処理失敗として処理を終了する。

以上、使用者101によってスクリプト実行を要求された場合の処理

について述べたが、前述したとおり、スクリプト 1 0 2 から他のスクリプト 1 0 2 を呼び出すことも可能である。この場合、呼び出し元が異なるだけで、スクリプト制御部 2 0 3 に渡すデータ及びその後の処理は同一である。

5 次に、図 2 を参照しながらプラグイン制御部 2 0 4 の動作について説明する。プラグイン制御部 2 0 4 は、スクリプト 1 0 2 の検査手順記述部 4 0 5 に記述されたプラグイン実行命令をスクリプト制御部 2 0 3 が実行した時にスクリプト制御部 2 0 3 によって呼び出される。呼び出し時に渡されるデータは実行するプラグイン 1 0 4 の名前及びそのプラグイン 1 0 4 が必要とする実行パラメータである。

10 プラグイン制御部 2 0 4 はプラグイン蓄積部 2 0 7 から、パラメータとして渡されたプラグイン名に対応するプラグイン 1 0 4 を取り出して実行する。実行結果は呼び出し元であるスクリプト制御部 2 0 3 に返され、最終的にはプラグイン実行命令に対する結果としてスクリプト 1 0 2 に戻される。

15 プラグイン 1 0 4 はその実行中に、踏み台模擬プログラム制御部 2 0 5 を通じて、踏み台模擬プログラム 1 0 5 を操作する。操作される踏み台模擬プログラム 1 0 5 は、プログラムの動作しているホストコンピュータのアドレスと、ホストコンピュータ内部でユニークな踏み台模擬プログラム識別子で指定される。踏み台模擬プログラム 1 0 5 に要求できる命令は以下の通り。

TCP/UDP/RAWソケット生成・破棄
ソケット (TCP/UDP) のローカルポートへの Bind
ソケット (TCP/UDP) のリモートポートへの Connect
20 Connectされたソケットを通じた Send、Recv
Connectされていないソケットを通じた SendTo、Rec

v F r o m

P r o c e s s の起動・終了

起動した P r o c e s s の標準入出力を通じたデータのやり取り

脆弱性検査装置ホストから踏み台模擬プログラム動作ホストへのファイ

5 ル転送及びその逆踏み台模擬プログラム状態取得

踏み台模擬プログラム停止

次に図 2 を参照しながら知識共有部 1 0 3 の動作について説明する。

知識共有部 1 0 3 は知識蓄積部 2 0 8 に、検査によって得られた知識を蓄積し、他の検査でそれを再利用することを可能にするために使用される。

10

推論部 1 0 8 は、与えられたゴールを満たす解が存在するかどうか、知識蓄積部 2 0 8 中の知識に基づいて推論を行う。本手段は、スクリプト 1 0 2 の実行条件の確認のためにスクリプト制御部 2 0 3 によって呼び出される。また、スクリプト 1 0 2 に共有知識獲得命令を記述しておくことで、スクリプト実行中に呼び出される場合もある。

15

知識は述語論理で表現されており、推論は P r o l o g 等の、述語論理に基づいた推論システムによって行われる。知識蓄積部 2 0 8 には、検査で得られた事実に関する知識だけでなく、変数を利用した推論ルールも蓄積しておくことが可能である。

20

また、スクリプト 1 0 2 を実行する作用を持った特別な述語が定義されており、この述語を利用した推論ルールを記述しておくことで、共有知識が不足の場合に知識を獲得するためにスクリプト 1 0 2 を実行することができる。これにより、あるスクリプト 1 0 2 の実行条件を満たすために、自動的に他のスクリプト 1 0 2 を呼び出すことが可能となる。

25

推論ルールは通常システム初期化時に初期設定ファイル（知識ファイル）から読み取られ、共有知識蓄積部 2 0 8 に設定されるが、検査の過

程で追加することも可能である。また、蓄積された知識を初期設定ファイル（知識ファイル）に保存することも可能である。

知識ファイルの例を図7に示した。本実施の形態では、記法はPrologの文法を利用している。

- 5 本実施の形態で示されるシステムにより、次のような特徴を持ったセキュリティホール診断システムを実現することができる。

10 第一に検査シナリオを、プログラミング言語で記述されたスクリプト102として表現し、スクリプト102から自動的にプラグイン（検査実行部に該当）104を呼び出すことで、複雑な試験の実施を実施できる。

さらに、各検査実行部間のパラメータの授受はスクリプト102が媒介するようにすることで、使用者は検査実行部間の入出力の関係について知っている必要を無くせる。

- 15 さらに、スクリプト102が他のスクリプト102を呼び出せるようにすることで、階層化されたシナリオの実施を実現できる。

さらに、推論ルールに従って、共有された知識から新たな知識を導出できるようにすることで各スクリプト102・プラグイン104毎に推論ロジックを作りこむ必要がなくなる。

- 20 さらに、プラグイン104が、踏み台模擬プログラム105を経由して検査を実行することで、現実の攻撃者と同様な踏み台を経由した検査シナリオを実現できる。

- 25 さらに、スクリプトにクラス概念の概念を採り入れることにより、クラス名によるグループ分けが可能になり、スクリプトから他のスクリプトを呼び出すときに、スクリプトのファイル名でなく、クラス名からも呼び出すことができる。

実施の形態 2.

実施の形態 1 では、操作部 2 0 1 と検査実行部 2 0 2 は同一装置内に存在するが、これらをネットワーク上に分散配置することも可能である。

- 5 本実施の形態で示されるシステムにより、次のような特徴を持ったセキュリティホール診断システムを実現することができる。

実施の形態 1 における特徴に加え、検査実行部をファイアウォールの外側に配置し、操作部をファイアウォールの内側に配置することが可能となり、これにより、本システムをネットワーク上に配置することのセキュリティ上のリスクを低減することが可能である。

10

実施の形態 3.

実施の形態 1 では、プラグイン 1 0 4 として動的ロード可能な共有ライブラリを用いているが、踏み台模擬プログラム制御部 2 0 5 とのインタフェースを提供可能なインタプリタ言語によっても実現可能である。

15

本実施の形態で示されるシステムにより、次のような特徴を持ったセキュリティホール診断システムを実現することができる。

実施の形態 1 における特徴に加え、プラグイン 1 0 4 をより実装しやすくなる上、システム運用中でも簡単にプラグイン 1 0 4 を編集可能となる。

20

実施の形態 4.

本実施の形態では踏み台模擬プログラム 1 0 5、1 0 6 同士、及び踏み台模擬プログラム 1 0 5 と脆弱性検査装置 1 0 0 との間の通信は TCP / IP 上の独自プロトコルを用いたが、ファイアウォールを考慮してこれを HTTP、SMTP 等のファイアウォール通過可能な一般的通

25

信プロトコル上に構築することも可能である。

本実施の形態で示されるシステムにより、次のような特徴を持ったセキュリティホール診断システムを実現することができる。

- 5 実施の形態 1 における特徴に加え、踏み台模擬プログラムとの通信がファイアウォールによって遮断されることを防ぐことができ、より実際の攻撃者と同等の攻撃シナリオで検査を行えるようになる。

産業上の利用可能性

- 10 以上述べたように本発明によれば、検査シナリオを、プログラミング言語で記述されたスクリプトとして表現し、スクリプトから自動的にプラグイン（検査実行部に該当）を呼び出すことで、複雑な試験を実施できる。

- 15 さらに、各検査実行部間のパラメータの授受はスクリプトが媒介するようにすることで、使用者は検査実行部間の入出力の関係について知っている必要を無くせる。

請求の範囲

1. 不正アクセスのために通常攻撃者が行う手順がプログラミング言語で記述されたスクリプトが複数蓄積されたスクリプト蓄積部
5 と、

利用者からの入力により上記スクリプトの一覧を要求する操作部と、
上記操作部の要求に応じ、上記スクリプト蓄積部から各スクリプトを
取り出し、入出力パラメータ記述、スクリプト実行必要条件、検査手順
を表示したリストを作成して利用者に提示し、利用者が選択したスクリ
10 プトを実行するスクリプト制御部と、

個々のセキュリティホール攻撃のためのロジックが実装されたプラグ
インが蓄積されたプラグイン蓄積部と、

スクリプト制御部がスクリプトを実行することにより呼び出され、上
記プラグイン蓄積部から実行スクリプトが指定するプラグインを取り出
15 して、そのプラグインを検査対象コンピュータに対して実行するプラグ
イン制御部
とを備えたセキュリティホール診断システム。

2. パケット送受信、プロセス起動・終了・プロセスとのデ
ータ入出力、ファイル転送機能を有する踏み台模擬プログラムと、上記
20 プラグインからの指令により検査対象コンピュータに対するプラグイン
の実行を上記踏み台模擬プログラムを介して実施する踏み台模擬プログ
ラム制御部とを備えたことを特徴とする請求項1記載のセキュリティホ
ール診断システム。

3. 上記スクリプトは他のスクリプトを呼び出せる機能を有
25 するように構成されたことを特徴とする請求項1に記載のセキュリティ
ホール診断システム。

4. 上記スクリプトにクラス概念が導入され、上記スクリプトは、他のスクリプトを呼び出す際、クラス名を指定することにより他のスクリプトを呼び出せる機能を有するように構成されたことを特徴とする請求項1記載のセキュリティホール診断システム。

5 5. 上記スクリプト実行必要条件が満たされているか否かを確認する知識共有部を備え、知識共有部は上記スクリプトが実行される過程で収集された情報を推論ルールに従って新しい知識に導出する推論部を有することを特徴とする請求項1記載のセキュリティホール診断システム。

10 6. 知識共有部は、共有知識が不足の場合に、推論ルールに従って知識獲得のためのスクリプトを実行する機能を有するように構成されたことを特徴とする請求項5記載のセキュリティホール診断システム。

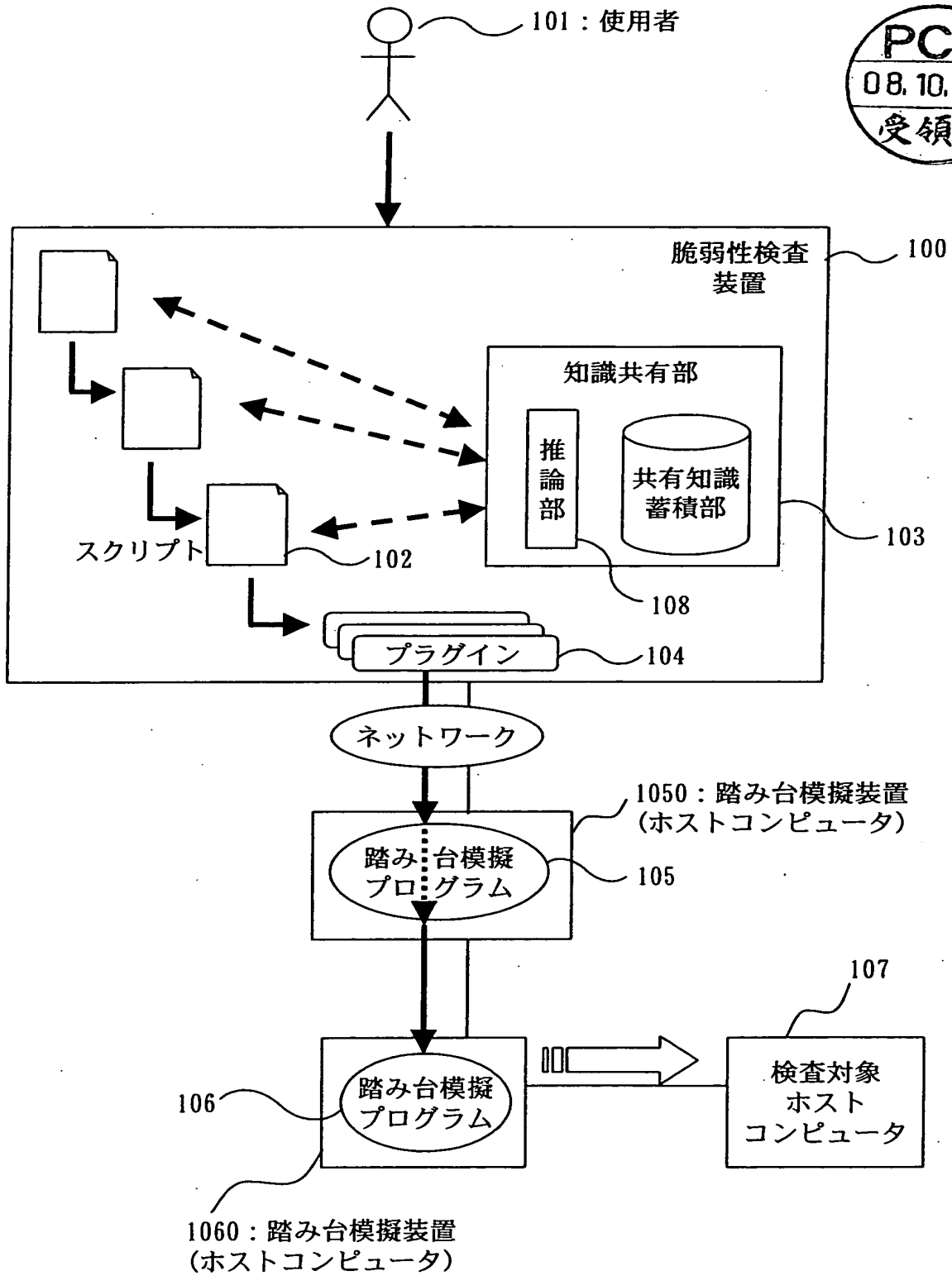
15 7. 上記スクリプト制御部と、上記プラグイン蓄積部と、上記プラグイン制御部と、上記スクリプト蓄積部と、上記踏み台模擬プログラム制御部とで検査実行部を形成し、検査実行部と上記操作部とはネットワーク上に分散した構成とされた請求項2記載のセキュリティホール診断システム。

20 8. 上記プラグインはインタプリタ言語で記述されたものであることを特徴とする請求項1記載のセキュリティホール診断システム。

9. 踏み台模擬プログラム制御部は、ファイアウォールが通過可能なプロトコル上で構築されていることを特徴とする請求項2記載のセキュリティホール診断システム。

要 約 書

- 通常攻撃者が行う手順をプログラミング言語で記述されたスクリプトが予め蓄積され、その蓄積されたスクリプトから使用者が選択したスクリプトを実行することにより、個々のセキュリティホール攻撃のための
- 5 ロジックが実装されたプラグインを呼び出し、このプラグインを検査対象コンピュータに対して実行することで、使用者は検査実行部間の入出力の関係等、セキュリティ上の知識を知っている必要が無くせる。



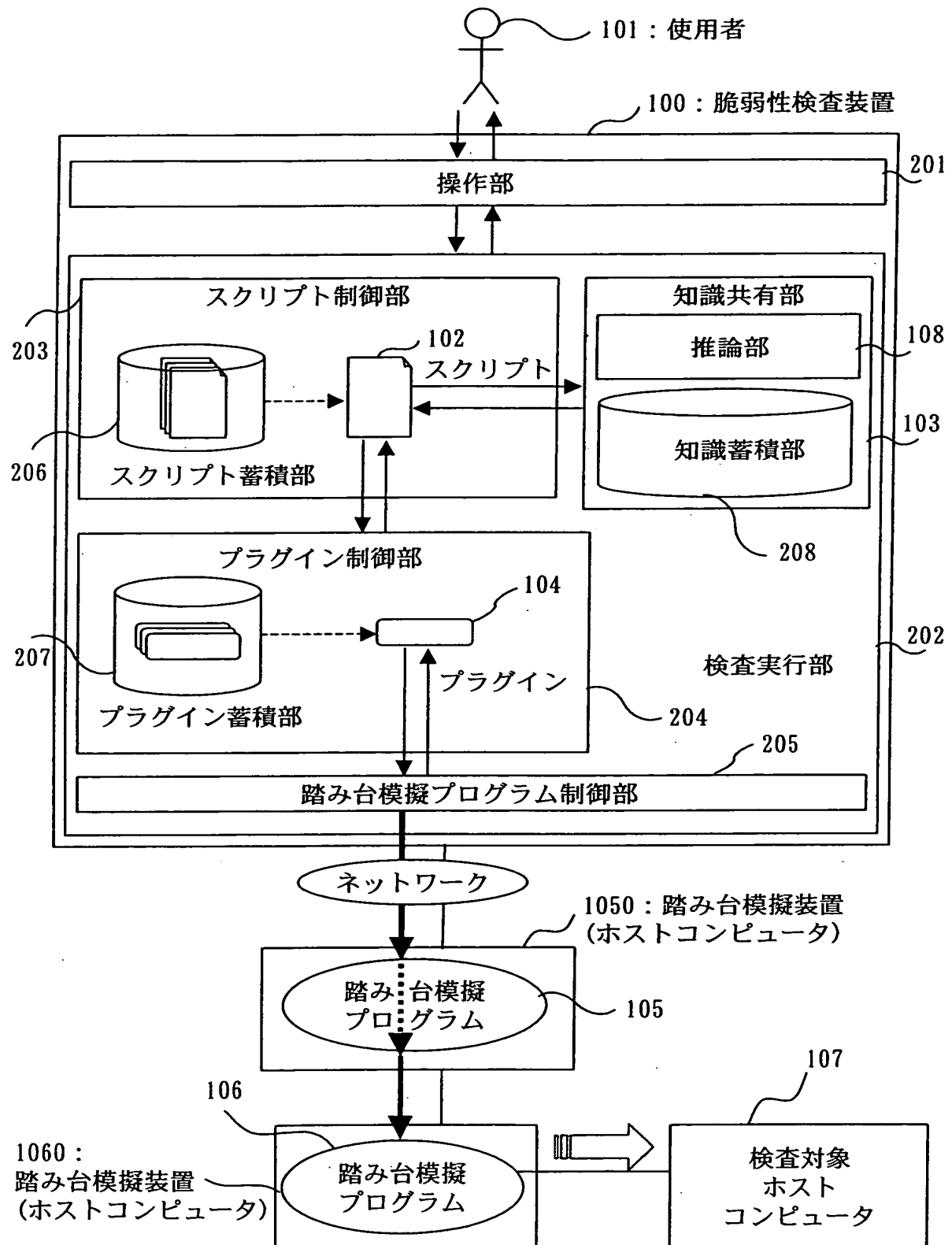


図 3

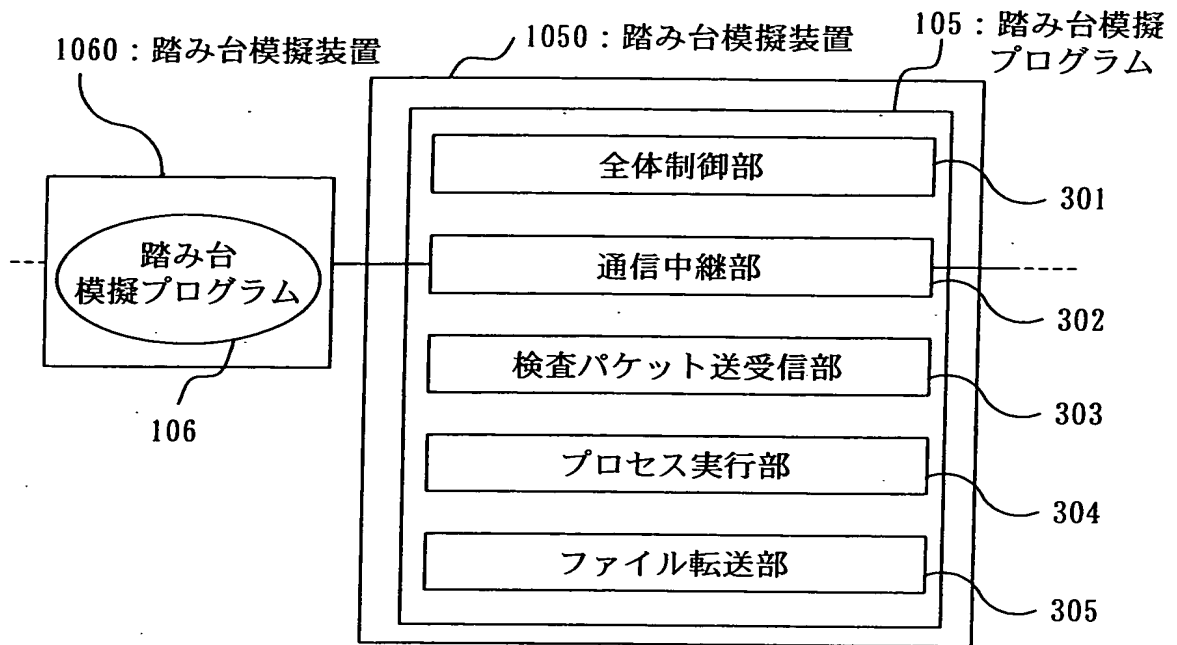
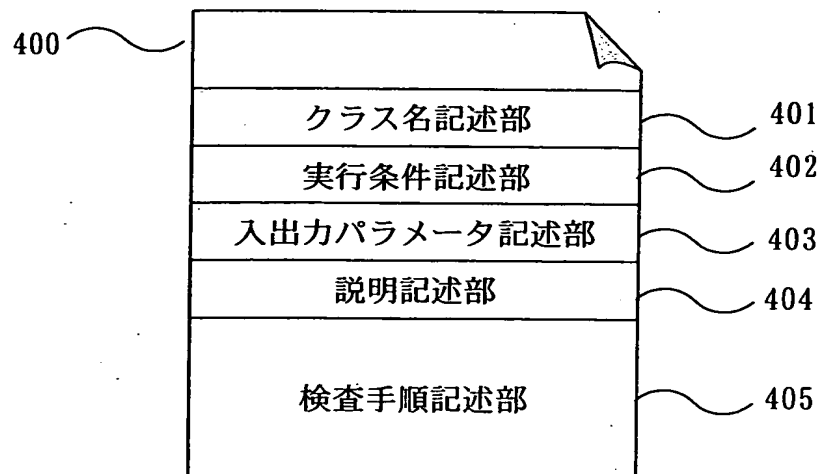
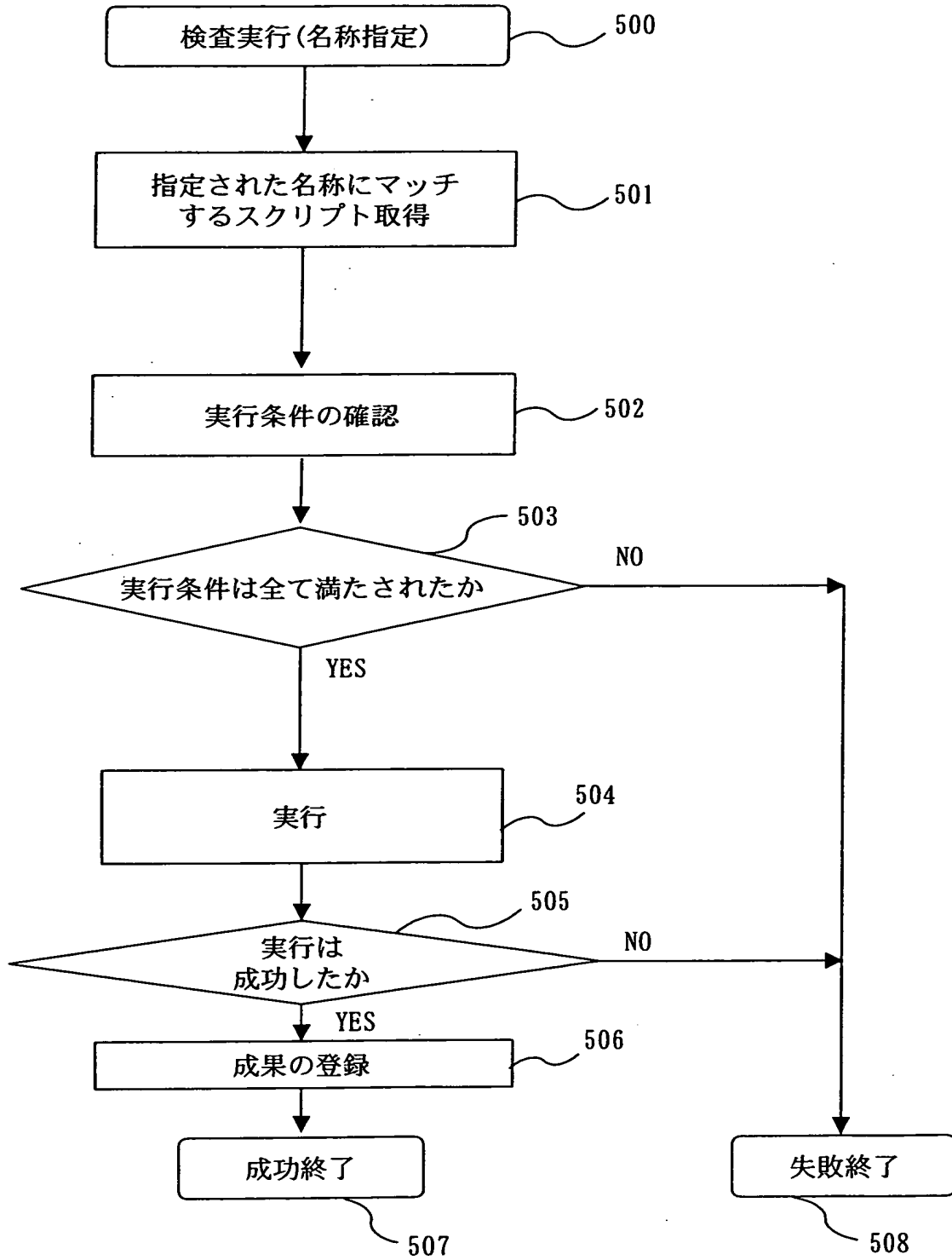
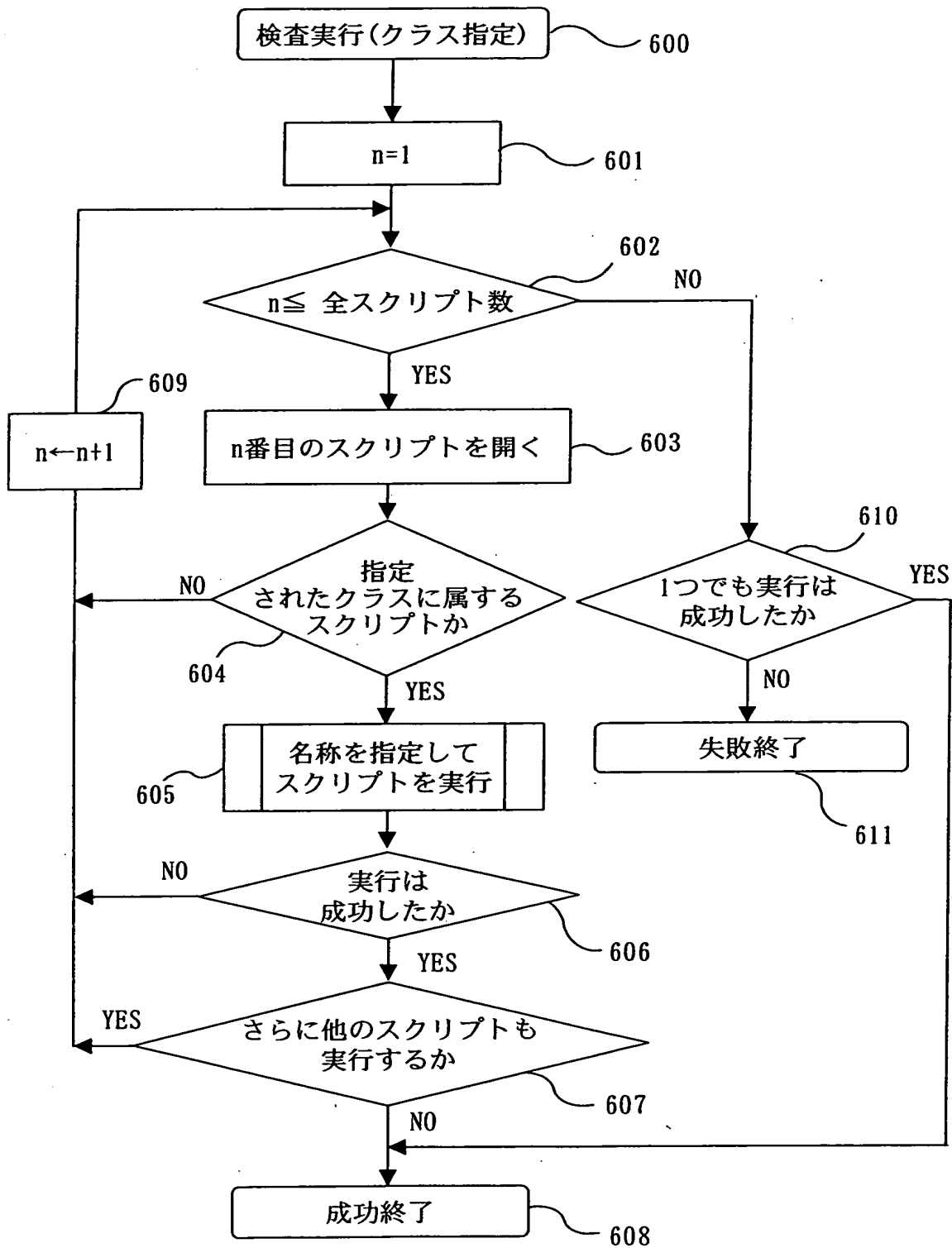


図 4







```

admin( HOST, 'root' ) :-
    os( HOST, 'UNIX' );

admin( HOST, 'administrator' ) :-
    os( HOST, 'WINDOWS' ).

web( HOST, PORT ) :-
    script( 'find_web_port', [ HOST ], [ PORTS ] ),
    member( PORT, PORTS ).

cgi( HOST, PORT, CGINAME ) :-
    script( 'check_cgi', [ HOST, PORT, CGINAME ], _ ).

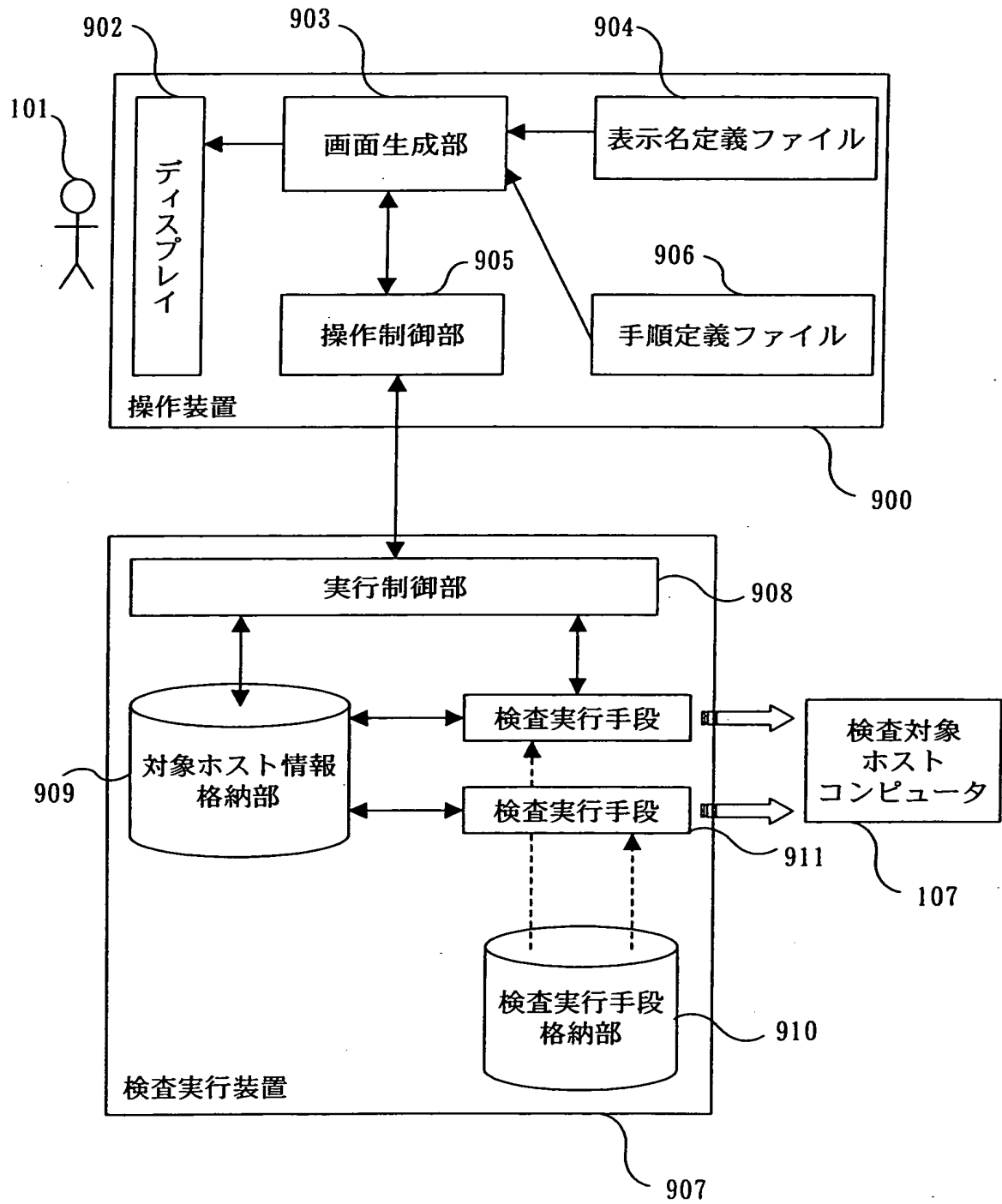
```

```

# -----BEGIN_SCRIPT_PROPERTY-----
#Class:      command_exec
#Pre-condition:  web( HOST, PORT ), cgi( HOST, PORT, "phf" )
#Input:      HOST      string  "検査対象ホストアドレス"
#Input:      CMD       string  "実行するコマンド文字列"
#Output:     RESULT    string  "コマンド実行出力"
#Description: "PHFを利用してコマンドを実行します。"
#-----END_SCRIPT_PROPERTY-----

Perlによる実行コード. . .

```



```
# # 手順定義 File.
# #
# # PROPERTY NAME
# # PLUGIN_TYPE
# #
# # SYNTAX: CATEGORY_KEY=' CATEGORY_DISPLAY_NAME(' EXEC_TYPE(' CATEGORY_DESCRIPTION)
# #          EXEC_TYPE = 0(ne)/A(11)/Q(query)
# #
# # PORTSCAN=ポートスキャン Q 対象ホスト上の攻撃可能なポートを調べます。
# # GET_FILE=ファイル取得 O ターゲットから指定されたファイルを取得します。
# # PASSWORD_CRACK=パスワードクラック Q 指定されたパスワードファイルからパスワードとパスワードを抽出します。
```

